Vol. 34 No. 5　　情　報　処　理　　May 1993

解　説

アナロジー

4. 抽象化に基づく類推†

桜井成一朗††　脇園竜次†††　原尾政輝††††

1. まえがき

類推は，対象となる問題の解決に既存の知識を利用する高次推論の一つであり，計算機科学，人工知能，認知科学などの分野において，活発に研究が行われている[7]．類推では，変換の源となる知識はソースと呼ばれ，変換後の知識はターゲットと呼ばれる．また，ソース領域からターゲット領域に直接的に写像される類推と，抽象的な領域を介して間接的に写像される類推とに大別することができる．抽象化に基づく類推は，抽象化あるいは抽象化図式としての知識を仮定し，既存の抽象化の具体化として解が得られるので，抽象化によって探索空間を適切に絞り込むことができる．本稿では，抽象化に基づく類推についてさまざまな観点から解説を試みる．

本稿は以下のように構成される．まず，**2.** でGreiner の抽象化に基づく類推について概説し，**3.** では抽象化に基づく類推証明について計算機科学の立場から解説し，**4.** では抽象化による誘導と類推について，探索空間縮小のための手法としての立場から解説する．**5.** では抽象化に基づく類推を基礎とした抽象化による学習について解説し，**6.** で認知科学的な観点から解説する．**7.** で本稿のまとめを行う．

2. Greiner の抽象化に基づく類推

Greiner[6] は，抽象化を仮定して，効率的で“有用な類推”の実現について述べている．

2.1 有用な類推の定義

類推システムにおける，領域知識と類推による予測の関係は**図-1** に示される．

図-1 はY字型に結合されたパイプ中を流れる流体の流量を求める問題である．類似領域である電気回路に関する豊富な知識が与えられているのに対して，流体系に関する知識が不足しているので，流体の流量を求められない．したがって，不足している知識，この場合，流体のキルヒホッフ則 (R_{K1Q}) を電気回路に関する知識から予測することが類推システムの課題となる．類似領域の知識から予測される仮説は少なくないので，**図-2** の類推演算子 $|\sim$ によって有用な類推が定義された．類推演算子 $|\sim$ では，A及びBを2階の述語変数として，領域理論 Th，ヒント $A \sim B$，問題 PT が与えられたときに，命題 $\varphi(A)$ が予測として出力される．図-1 では，Bは電流 (Current)，Iであり，Aは流量 (Flowrate)，Q であり，PT は“流量を求める”である．ヒントは“Current～Flowrate”であり，“電流”と“流量”が類似していることを表す．類推システムには，x を2階の述語変数とする抽象式 $\varphi(x)$ があらかじめ与えられており，抽象式は抽象化 (abstraction) とも呼ばれ，この抽象式によってソースとターゲット間の効率的な対応付けが可能となる．問題解決に必要な $\varphi(A)$ が，領域理論とヒント及び問題からの予測として類推演算子 $|\sim$ により得られ，図-1 では R_{K1Q} で表される．

学習は，領域理論 Th の演繹的閉包の拡張，すなわち，$Th' = Th \cup \{\rho\}$ の構成として捉えることができ，Dietterich は演繹的閉包に含まれない知識 $\{\rho\}$ の学習を“知識レベルの学習[3]”と呼んだ．類推演算子 $|\sim$ では，Unknown 制約によって $\varphi(A)$ が対象領域の知識から演繹されないことを保証するので，$|\sim$ によって得られる予測 $\varphi(A)$ を領域理論 Th に追加することによって知識レベルの学習が可能となる．また，$\varphi(A)$ が追加さ

† Abstraction Base Analogy by Seiichirou SAKURAI (Tokyo Institute of Technology), Ryuji WAKIZONO (Heavy Apparatus Engineering Lab., Toshiba Corp.) and Masateru HARAO (Kyushu Institute of Technology).

†† 東京工業大学総合理工学研究科
††† (株)東芝重電技術研究所
†††† 九州工業大学情報工学部

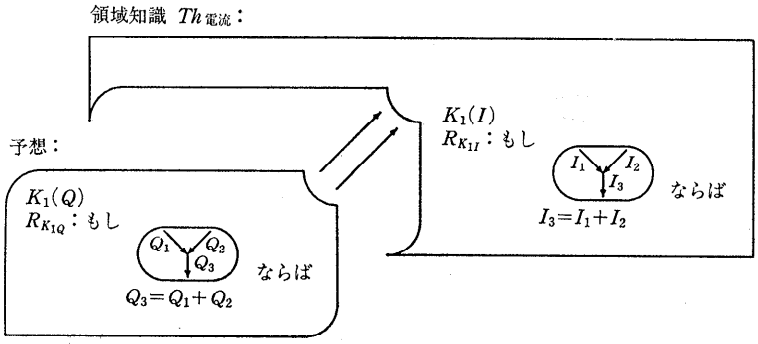


図-1 領域知識と類推による予測の関係[6]

$$Th, A\sim B \mid\sim_{PT}\varphi(A)$$

ただし，
Unknown: $Th \not\models \varphi(A)$
Consistent: $Th \not\models \neg\varphi(A)$
Common: $Th \models \varphi(B)$
Useful: $Th \cup \{\varphi(A)\} \models PT$

図-2 有用な類推の定義

れた領域理論の無矛盾性が Consistent 制約によって保証される．類似性は2階の類推式 φ によって捉えられ，φ が共通性質であることは Common 制約によって保証される．類推式が目標の問題を解くのに役立つことを保証するのが，Useful 制約である．

2.2 抽象化に基づく類推

Greiner の類推システムでは，ソースとターゲットの共通の抽象的知識，すなわち抽象化を仮定することが，従来の類推システムの多く[7]と異なっている．すなわち，抽象化に基づく類推では，**図-3** に示すような抽象化に基づいてソースとターゲット間の写像が行われるのである．図中，t, c, r, l はそれぞれ2階の述語変数であり，抽象的なキルヒホッフの第一則及び第二則，オームの法則がそれぞれ $K_1(t)$, $K_2(c)$, $Ohms(t,c,r,l)$ により表されている．

このような抽象化に対して，図-1 のように類似領域として電気回路に関する知識が与えられているときには，**図-4** に示すようなソース具体化

$$RKK(t,c,r,l) \Leftrightarrow \left\{ \begin{array}{c} K_1(t) \\ \& \\ K_2(c) \\ \& \\ Ohms(t,c,r,l) \end{array} \right\}$$

図-3 RKK 抽象化の定義

K_1(電流)：端子に流入する電流の総和は0である．
K_2(電圧降下)：閉路の電圧降下の総和は0である．
$Ohms$(電流，電圧降下，抵抗値，抵抗器)：
オームの法則

図-4 RKK のソース具体化

が得られる．すなわち，t, c, r, l はそれぞれ“電流”，“電圧降下”，“抵抗値”，“抵抗器”に具体化されて，電気回路におけるキルヒホッフの第一則及び第二則，オームの法則が得られる．

図-4 のソース具体化が得られると，ターゲット具体化として

RKK（流量，圧力降下，パイプ特徴，パイプ）

という流量に関する命題が予測される．予測された流体のキルヒホッフの法則を適用すれば，パイプ中を流れる流体の流量を求めることができる．

抽象化が与えられない場合には，以下のようなソースとターゲット間の対応付けを求めることもできるであろう．

電流 ↔流量
電圧降下↔圧力降下
抵抗値 ↔パイプ特徴
⋮

しかしながら，ソースとターゲットの記述が増加すれば，対応付けの候補も組合せ的に増加していく．これに対して，抽象化を利用した場合には，抽象的な関係が共通に成立することを仮定することによって，対応付けの候補を絞り込むことができる．

Greiner は抽象化に基づく類推システムを生成-検査法により実現し，さまざまなヒューリスティックスを導入することにより効率的な問題解決が可能となることを実験的に示した．ヒューリスティックスの詳細については参考文献 6）を参照されたい．

3. 抽象化に基づく類推証明

抽象化を制約付きの高階論理式で表現することによって，証明の非決定性を軽減することができる[8]．抽象化に基づく類推証明[8]では，類推のソースとして証明が与えられ，ソースの証明から高階論理式を抽象化として抽出し，抽象化からターゲットの証明が得られる．Greiner の類推演算子と比較すれば，A 及び B がターゲットとソースの証明に対応し，証明から抽出された高階論理式が φ に対応付けられるであろう．

3.1 LK システム

抽象化に基づく類推証明で扱う対象は，古典論理の論理式であり，証明系として LK を用いる．LK では，シーケントと呼ばれる $\Gamma \Rightarrow \Delta$ という形式の式が扱われる．ただし，Γ 及び Δ は式の有限集合であり，シーケントが真になるのは，Γ がすべて真のときに Δ 中のどれかの式が真のときである．論理式は以下のように帰納的に定義される．ただし，A はアトムであり，～ は否定を表し，∨は選言を，∧は連言を，⊃は論理的含意を表す．

$$\phi ::= A \mid \sim\phi \mid \phi \vee \phi \mid \phi \wedge \phi \mid \phi \supset \phi \mid \forall x.\phi \mid \exists x.\phi$$

シーケントの中で，$A \Rightarrow A$ は自明なシーケントであり，これを公理と呼ぶ．LK の推論規則については紙面の都合で割愛する．LK の証明は，推論規則の非決定的適用により作られ，導出木で表される．導出木の葉 (leaf) のラベルがすべて公理でラベル付けされているならば，その根にある論理式は証明可能であると言う．

図-5 に論理式，$\phi = \Rightarrow (p(a) \vee q(b)) \wedge \forall x(p(x) \supset q(x)) \supset \exists x q(x)$ の LK の証明の例を示すが，証明は非決定的に構成されるため，必ずしも効率的ではない．

3.2 証明の類似性とスキーマ

証明の類似性は型の概念[1]を用いて定義される．同一の証明構造をもつような論理式のクラスは，スキーマと呼ばれる 2 階の論理式によって特性化される．すなわち，ここで述べる手法は，同一の証明構造をもつような論理式のクラスに対して有効となる．たとえば，図-5 の ϕ と同一の証明構造をもつ論理式を特性化するスキーマは，$\Phi = [P(a) \vee Q(b)] \wedge \forall x.(P(x) \supset Q(x)) \supset \exists x.Q(x)$ と表せる．ただし，P, Q は 2 階の述語変数である．そのとき証明は**図-6** のように与えられる．

$$\dfrac{\dfrac{\dfrac{p(a) \Rightarrow p(a)}{p(a), (p(a) \supset q(a)) \Rightarrow \exists x q(x)}}{p(a), \forall x(p(x) \supset q(x)) \Rightarrow \exists x q(x)} \quad \dfrac{\dfrac{q(a) \Rightarrow q(a)}{q(a) \Rightarrow \exists x q(x)}}{q(b), \forall x(p(x) \supset q(x)) \Rightarrow \exists x q(x)}}{\dfrac{\dfrac{p(a) \vee q(b), \forall x(p(x) \supset q(x)) \Rightarrow \exists x q(x)}{(p(a) \vee q(b)) \wedge \forall x(p(x) \supset q(x)) \Rightarrow \exists x q(x)}}{\Rightarrow (p(a) \vee q(b)) \wedge \forall x(p(x) \supset q(x)) \supset \exists x q(x)}}$$

図-5 LK 証明

$$\dfrac{\dfrac{\dfrac{P(a) \Rightarrow P(a)}{P(a), (P(a) \supset Q(a)) \Rightarrow \exists x Q(x)}}{P(a), \forall x(P(x) \supset Q(x)) \Rightarrow \exists x Q(x)} \quad \dfrac{\dfrac{Q(a) \Rightarrow Q(a)}{Q(a) \Rightarrow \exists x Q(x)}}{Q(b), \forall x(P(x) \supset Q(x)) \Rightarrow \exists x Q(x)}}{\dfrac{\dfrac{P(a) \vee Q(b), \forall x(P(x) \supset Q(x)) \Rightarrow \exists x Q(x)}{(P(a) \vee Q(b)) \wedge \forall x(P(x) \supset Q(x)) \Rightarrow \exists x Q(x)}}{\Rightarrow (P(a) \vee Q(b)) \wedge \forall x(P(x) \supset Q(x)) \supset \exists x Q(x)}}$$

図-6 Φ の証明構造

1 階の論理式 ϕ と ψ が共に高階単一化によりスキーマ Φ のインスタンスであるとき，ϕ と ψ はスキーマ Φ の下で類似しているという．インスタンスの判定には高階単一化を用いているので，アトムだけでなくアトム以外の論理式間の類似性も扱うことができる．高階単一化についてはたとえば参考文献 14)，19) を参照されたい．

3.3 スキーマの構成

単一化理論は，与えられた項 t_1, t_2 について $\sigma(t_1) = \sigma(t_2)$ となるような代入 σ の存在を見つける問題に関係する．単一化の双対 (dual) は一般化あるいは反単一化，逆単一化 (unti-unification) などと呼ばれる．一般化は，項 s, t について s も t も Z のインスタンスとなるような項 Z を見つけることであり，この Z を s と t の一般化式と呼ぶ[10]．

ソースの論理式を g，対象問題の論理式を h とする．

$$g = [p(a) \vee q(b)] \wedge \forall x(p(x) \supset q(x)) \supset \exists x q(x)$$

$$h = ((p(a) \wedge r(a)) \vee (q(b) \wedge r(b))) \wedge \forall x(p(x) \supset q(x)) \supset \exists x q(x)$$

ここで，g と h が同一の証明構造をもつと仮定して，反単一化アルゴリズム[8]により次の制約付きスキーマを得ることができる．

$$schema_g = [\Phi(a) \vee \Psi(b)] \wedge \forall x(P(x) \supset Q(x)) \supset \exists x Q(x)$$

$$constraints = \Phi(a) \Rightarrow P(a), \Psi(a) \Rightarrow Q(b)$$

このスキーマの証明と制約条件の証明の融合によって，対象問題の証明が得られる．

3.4 類推を用いた証明

以下に与える類推を用いた自然演繹証明手続き

では，スキーマの利用により，非決定的操作を減らすことができる．

1. スキーマSを選択する．適用可能なスキーマがなければ終了する．
2. 対象問題wとSのマッチングを行う．($\sigma(S)=w$ なる σ を求める．) 失敗すれば1に戻る．
3. σが制約を満足することを確かめる．

失敗すれば1に戻り，成功すればSの証明と制約の証明を融合した証明を出力し終了する．

抽象化に基づく類推証明では，Greiner の手法と同様に抽象化としての共通な構造がソースとターゲットで保持されるということを仮定することによって効率化が図られている．解決すべき重要な問題点としては，スキーマ表現と証明の対応付けを洗練することと，一般的過ぎるスキーマの扱いがあげられる．また，この手法では，証明は得られるものの，最も適切な証明であることは保証されないので，得られた証明をより適切な証明に置き換えることも興味深い．

4. 抽象化による推論の誘導と類推

本章で述べる抽象化は，ターゲット領域の抽象化により抽象的な領域を自ら形成するので，ソース領域が不必要となる点で他の抽象化に基づく類推と異なっている．ここで述べる方法は，抽象化された領域から推論の本質部分を抽出し，ターゲット領域での推論を誘導することができる（**図-7** 参照）．

4.1 Plaisted の抽象化[11]

Plaisted の抽象化は，問題空間を表す言語の節を別の言語の節（集合）に変換する．節中にリテラルの重複を許した m(multi)-clause を対象とした，m-abstraction φ を考える．m-clause 間の導出を m-resolution と呼び，m-resolution による導出形を m-resolvent と呼ぶ．たとえば，$\neg p \vee \neg p \vee \neg p$ と p の m-resolvent は，$\square$, $\neg p$, $\neg p \vee \neg p$ である．以下では，m-resolution を単に導出と呼ぶことにする．m-abstraction は，m-clause を m-clause（の集合）に変換する．ただし，空節は空節に変換され，C_3 が C_1 と C_2 の m-resolvent，$D_3 \in \varphi(C_3)$ であれば，$D_1 \in \varphi(C_1)$ と $D_2 \in \varphi(C_2)$ が存在し，D_3 は，D_1 と D_2 の m-resolvent のインスタンスになるという条件を満足する．

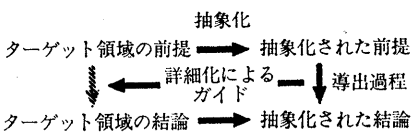


図-7 抽象化を利用した解の探索

m-clause の集合 S から C が導出されると，$\varphi(S)$ から $C' \in \varphi(C)$ が導出され，二つの証明木は同型になる．抽象化空間での証明木は元の証明木よりも単純であることが期待でき，これを証明のガイドとすることが考えられる．その手続きを与えるために，述語の引数を落とす“命題 m-abstraction”を例とする．$S=\{\neg P(a) \vee \neg P(b) \vee Q(c), P(a), P(b) \vee R(d)\}$，$C=Q(c) \vee R(d)$ とし，φ を命題 m-abstraction とすれば，$\varphi(S)=\{\neg P \vee \neg P \vee Q, P, P \vee R\}$，$\varphi(C)=\{Q \vee R\}$ となる．

基本手続きは，抽象化された空間での $\varphi(S)$ から $Q \vee R$ への証明木を作成し，その木の形をまねて元の空間での証明木を作る．この操作を，木の深さ0から始めて正しい証明木が得られるまで続ける．深さ0と1では正しい証明木は得られないが，深さ2では**図-8**・Step 1 のような証明木が得られる．

次に，この証明木から不要な部分を取り除く．ここで不要なものは，導出不可能な深さ1の $Q \vee R$ と，深さ2にある $Q \vee R$ を導くのに無関係な導出節である．すると**図-8**・Step 2 のようになる．

最後に，上の証明木にそってSからCへの証明木を作成したものが**図-9** である．

元の空間では，抽象化された証明木に対応する証明木だけを考えればよい．

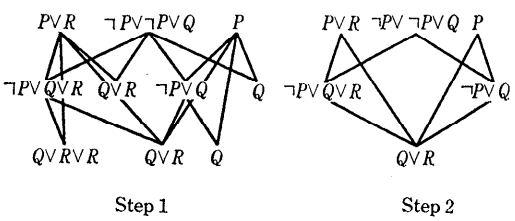


図-8 抽象化された証明木

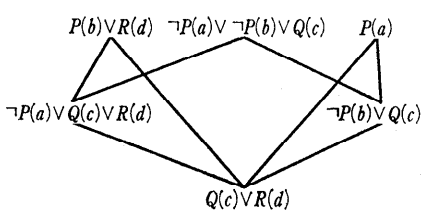


図-9 導かれた証明木：Step-3

4.2 Tenenberg の抽象化[15),16)]

Tenenberg は，述語記号の変換を行う述語写像（述語記号以外は自分自身へ写される）を用いた抽象化を提案した．Tenenberg の方法も，Plaisted の方法と同様に具体レベルの問題を抽象空間に写像するので，ソースを必要としない．Plaisted の方法が，具体レベルの証明を構築するためのガイドとして抽象レベルの証明を利用しているのに対して，Tenenberg の方法では抽象レベルでの問題解決過程が直接具体レベルに引き戻されて，具体レベルの問題が解決されるという点で異なっている．

具体レベルの述語記号を単純に抽象レベルの述語記号に写像すると，以下のような問題が生じる．たとえば，$\{Bottle(A), \neg Cup(A)\}$ に対して $Bottle \rightarrow Container$, $Cup \rightarrow Container$ なる述語写像を用いると，以下のように抽象化できる．

$\{Container(A), \neg Container(A)\}$

このように，無矛盾な具体レベルの理論を単純に抽象化すると抽象レベルの理論が矛盾してしまう場合がある．Tenenberg はこの問題点を解消し，理論の無矛盾性が保存されるような抽象化の概念をモデル理論及び証明論的観点から展開した．以下ではその概略を示す．

Tenenberg は，**図-10** に示す述語写像に対するモデルの抽象化の概念を与えた．

この図は，節集合 S の任意のモデル M の抽象化 M' が抽象化された節集合 S' のモデルとなることを要求している．ここでは，述語写像を用いたこの条件を満たす抽象化写像について考える．

全射である述語写像を f，節 C の負リテラルだけからなる節を $neg(C)$，正リテラルだけからなる節を $pos(C)$，節 C 中のリテラルの数を $|C|$ で表す．抽象化写像 $MembAb_f$ を以下に与える．

$MembAb_f(S)=$

$\{C'' \mid |neg(C')|$ 個のリテラルをもつすべての $N \in f^{-1}(neg(C'))$ に対し，$N \vee P \in S$ を満たす $P \in f^{-1}(pos(C'))$ が存在する$\}$

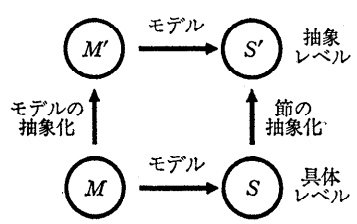


図-10 抽象化モデル

直観的には，負リテラルを含むような抽象化された節は，節の負リテラルの各具体化と正リテラルのある具体化から構成される節が必ず元の節集合に含まれていることが保証されている．たとえば，述語記号 $\{Bottle, Cup, Graspable, Madeof Glass, MadeofCeramic\}$ をもつ言語 L から述語記号 $\{Container, Graspable, Breakable\}$ をもつ言語 L' への述語写像 f を $\{Bottle, Cup\} \rightarrow Container$, $\{Graspable\} \rightarrow Graspable$, $\{Madeof Glass, MadeofCeramic\} \rightarrow Breakable$ とする．L の節集合 S を

$\{\neg Bottle(X) \vee Graspable(X),$
$\neg Cup(X) \vee Graspable(X), Bottle(a),$
$\neg Cup(X) \vee MadeofCeramic(X), \neg Cup(b)\}$

とすると，$\neg Bottle(b) \notin S$ かつ空節 $\square$ については $f^{-1}(\square)=\square$ であるので，$\neg Container(b)$ という抽象化は $MembAb_f(S)$ に含まれない．また，$\neg Container(X) \vee Breakable(X)$ も $\neg Container(X)$ の $\neg Bottle(X)$ への引き戻しに対する $Breakable(X)$ の引き戻しが存在しないので抽象化されない．したがって，$MembAb_f(S)$ は以下のようになる．

$\{\neg Container(X) \vee Graspable(X),$
$Container(a)\}$

S が無矛盾であれば，$S'=MembAb_f(S)$ も無矛盾であり，S' からの証明木 W' が存在すれば，$MembAb_f^{-1}(S') \subseteq S$ からの証明木 W が存在し，W に現れる各節は f により W' に現れる節と対応している．さらに，S をホーン節の集合，G' をアトムとすると $MembAb_f(S)$ から G' の証明木が存在すれば，それと同型の S から $G \in f^{-1}(G')$ への証明木が存在する．このことは，元の空間での証明木作成の制約となる．

抽象化には USP (Upward Solution Property) と DSP (Downward Solution Property) と呼ばれる二つの側面がある[15)]．USP とは，元の空間で解が存在すれば抽象化された空間でも解が存在することがであり，DSP は逆に，抽象化された空間で解が存在すれば元の空間でも解が存在することである．上記の Tenenberg の例は DSP に基づいたものであるが，USP に基づく研究もいくつ

かみられる[9),18)].

抽象化に関連したそのほかの研究では，Plaistedの抽象化において代入子を利用することを採用して，解の探索空間をより縮小できる参考文献 17) や類推におけるソース領域の探索空間の縮小を目指した参考文献 21) などもある．

5. 抽象化による学習

本章では，論理プログラムを対象とした，抽象化に基づく類推と帰納学習を組み合わせた抽象化による学習[12)]について解説する．ここでは，帰納学習の効率化のために，2階の論理プログラムとして与えられる抽象化が利用される．Greiner の手法と同様に，抽象化がソースとターゲットの共通構造であると仮定することによって，推論の効率化が図られる．

5.1 抽象化スキーマの具体化

抽象化スキーマは，**図-11** に示すような2階の論理プログラムとしてあらかじめ与えられる．

図-11 に示したスキーマは分割統治法のスキーマであり，抽象述語は2階の述語変数によって表現される．*Decomp* によって問題が分割され，*Aux* が分割した個々の問題が解決され，*Comp* で *Aux* の結果が合成されることを表す．また，*related_vars* は探索空間を絞り込むための，制約条件である．

5.2 抽象化に基づく学習

学習は，Greiner の枠組と同様に，ヒントを基にして，スキーマが検索され，スキーマの具体化として，ターゲットの論理プログラムが得られる．帰納学習では，十分一般的な知識を獲得するために，一般化や特殊化のさまざまな演算子が用いられるが，スキーマとして論理プログラムの骨格が与えられるので，演算子適用の効率化を図ることができる．また，スキーマとして骨格が与えられるということは，プログラム中に現れる述語の数が限定されるので，訓練例に含まれない述語記号の生成もでき，すなわち構成的な帰納推論が可能となる．

```
abs([S(T, R) :-
    Guard(T1), Decomp(T1, A),
    Aux(A1, B), Comp(B1, R)]) :-
    related_vars(T, T1),
    related_vars(A, A1),
    related_vars(B, B1)
```

図-11 抽象化スキーマの例

このような高階の抽象化スキーマによる学習の効率化の試みとしては参考文献 4) もある．

抽象化を利用した学習システムでは，十分な抽象化スキーマを与えることができれば，かなり強力な学習システムを構築できるものの，他の抽象化に基づくシステムと同様に，抽象化スキーマをどのように与え，またどのように組織化しておくかという課題が残されている．

6. 認知科学における抽象化に基づく類推

6.1 スキーマの帰納と類推[5)]

人間の類推的思考においては，具体的事例から一般的なスキーマの帰納が類推を可能にしていることが予測される．Gick と Holyoak[5)] はこの予測を確認するために，問題とその解を読んだ被験者に類似の問題を解かせるという実験を行った．この実験では，類推が特殊な概念のより一般的なスキーマへの写像及び具体的事例からのスキーマの帰納も含んでいるということが示唆される．

実験は類似の物語を被験者に読ませ，明示的なヒントを与えることなく放射線問題を解かせ，次に類似物語による示唆を利用して放射線問題を解かせることによって行われた[5)]．具体的問題とスキーマの例を**図-12** に示す．収束スキーマは，中心の対象に対して一斉に力を働かせることができないが，弱い力を同時に働かせることによって，目的を達成するスキーマである．

Gick と Holyoak[5)] は，単一の具体例からスキーマが帰納されるかどうかの実験と，複数の具体

	軍隊問題	放射線問題	収束スキーマ
問　題	軍隊による要塞捕獲 全軍移動不可能な道	放射線による腫瘍の破壊 一方向から強い放射線を照射できない.	力による中心的対象の克服 全力適用不可能
プラン	複数の道からの 小隊の同時攻撃	四方からの 弱い放射線照射	複数の弱い力の同時適用

図-12 収束問題とスキーマ

例からスキーマが帰納されるかどうかの実験を行った．前者の実験では，抽象的な表現が生成されるという証拠も得られず，類推に寄与するという証拠も得られなかった．これに対して，後者の実験からは肯定的な結果が得られた．二つの具体例が与えられたときに，被験者は副次的産物として収束スキーマを導く傾向があることが分かった．

6.2 人間の学習におけるプラグマティックな表現の役割[20]

鈴木・村山[20]は，ソース領域の表現がターゲット領域の表現に変換されるのは，各領域の対象や関係がある観点で同一視されるからであると考え，この観点を抽象化として捉えている．電流系と水流系の類推では，ともに「何かが流れる系」に抽象化され，個々の対象や関係も同時に抽象化されていると考えられる．たとえば，細いパイプ（抵抗）は「流れるものが通りにくいところ」に，水圧は「流れるものの勢い」に抽象化される．このような抽象化は，「流れ」に関する役割というプラグマティックな（すなわち目的的，機能的な）観点から抽象化されている．鈴木・村山[20]はプラグマティックな抽象化を準抽象化と呼び，共通属性の抽出による抽象化と区別している．これは，プラグマティックな観点が変われば，表現も変化すると考えられるが，単純に共通な属性を抽出したのでは，表現の変化を説明できないためであると考えられる．

初心者がオペレーティングシステムを学習する場合には，**表-1** の知識が用いられていると考えられる．

この実験で被験者の言い替えが頻繁に観察されるのは，被験者がコマンドとオペランドの組合せを「するもの-されるもの」という抽象的知識に準抽象化して学習しているからであると考えられる．「するもの-されるもの」とは，コマンドによってある目的を達成する観点からの分類であり，単純な共通属性の抽出とは区別される．この抽象的知識は基本知識であり，コマンドの構文を推論できるだけでなく，2)のフィルタ・コマンドの学習も比較的容易に行うことができる．これに対して，3) のリダイレクションの学習では，「入力-加工-出力」という知識が必要となる．したがって，基本知識だけで理解しようとする初心者は必然的に学習に時間がかかることになる．このような形式的知識の学習では，抽象的な既有知識への準抽象化プロセスを仮定すれば，学習過程に対する説明が与えられる．これは人間が抽象化に基づく類推により学習する場合があることを示していると考えられる．しかしながら，プラグマティックな準抽象化をどのように形式化していくかは今後の課題として残されている．

表-1 オペレーティングシステムの学習材料

1) 単純なコマンド	オペランドをとるOSの組込みコマンド
2) パイプ	フィルタ・コマンドを用いた連続処理
3) リダイレクション	入出力をデフォールト以外に変更する

7. あとがき

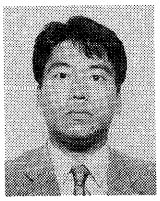
本稿では，抽象化に基づく類推について概説した．抽象化に基づく類推では，ソースとターゲットに共通の抽象化を仮定することにより，類推の効率化が期待できる．紙面の都合で触れられなかったが，抽象化に基づく類推による CBR[13] や類推の定式化を目指した参考文献 2) などがある．抽象化に基づく類推では，抽象化をどう与えるか，抽象化データベースをどのように構成するかという問題が残されている．

参考文献

1) Andrews, P. B. (小川原訳)：数理論理学とタイプ理論 証明による真理へ，丸善 (1987).
2) Deirbach, C. and Chester, D. L.: A Formal Basis for Analogical Reasoning, In *KR 91*, pp. 139-150 (1991).
3) Dietterich, T.G.: Learning at the Knowledge Level, *Machine Learning*, Vol. 1, No. 3, pp. 287-316 (1986).
4) Feng, C. and Muggleton, S.: Towards Inductive Generalization in Higher Order Logic, In *Machine Learning Workshop*, pp. 154-162 (1992).
5) Gick, M. L. and Holyoak, K. J.: Schema Induction and Analogical Transfer, *Cognitive Psychology*, Vol. 15, pp. 1-38 (1983).
6) Greiner, R.: Learning by Understanding Analogies, *Artificial Intelligence*, Vol. 35, pp. 81-125 (1988).
7) Hall, R. P.: Computational Approaches to Analogical Reasoning: A Comparative Analysis, *Artificial Intelligence*, Vol. 39, pp. 39-120 (1989).
8) Harao, M.: Abstraction Based Analogical Reasoning for Natural Deduction Proof, In *ILP*

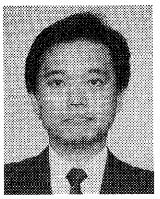
92, pp. 368-382 (June 1992).
9) Knoblock, C. A., Tenenberg, J. D. and Yang, Q.: Characterizing Abstraction Hierarchies for Planning, In *AAAI-91*, pp. 692-697 (1991).
10) Muggleton, S. and Feng, C.: Efficient Induction of Logic Program, In *ALT* 90 (1990).
11) Plaisted, D. A.: Theorem Proving with Abstraction, *Artificial Intelligence*, Vol. 16, pp. 47-108 (1981).
12) Sakurai, S. and Haraguchi, M.: Towards Learning by Abstraction, In *ALT '91*, pp. 288-298 (1990).
13) Shinn, H. S.: Abstractional Analogy: A Model of Analogical Reasoning, In *Proc. Case-Based Reasoning Workshop*, pp. 370-387, Morgan Kaufmann Pub. (1988).
14) Snyder, W. and Gallier, J.: Higher Order Unification Revisited, *Journal of Symbolic Computation*, Vol. 8, pp. 101-140 (1989).
15) Tenenberg, J. D.: *Abstraction in Planning*, Ph. D. thesis, University of Rochester, Dept. of Computer Science, NY (May 1988).
16) Tenenberg, J. D.: Inheritance in Automated Planning, In *Proc. of First International Conference on Principles of Knowledge Representation and Reasoning*, pp. 475-485 (1989).
17) Yamamoto, A.: An Anatomy of Abstraction, *Bulletin of Informatics and Cybernetics*, Vol. 22, No. 3-4, pp. 188-197 (1987).
18) Yang, Q. and Tenenberg, J. D.: ABTWEAK: Abstracting a Nonlinear, Least Commitment Planner, In *AAAI-90*, pp. 204-209 (1990).
19) 原尾，岩沼：高階ユニフィケーションアルゴリズムの複雑さについて，コンピュータソフトウェア，Vol. 8, No. 1, pp. 41-53 (1991).
20) 鈴木，村山：人間の学習におけるプラグマティックな表現の役割，日本認知科学会(編)，認知科学の発展，第4巻，pp. 79-103，講談社 (1991).
21) 脇園，有川，原口：類推のための抽象化，Technical Report SIG-FAI-8904-2，人工知能学会研究会資料 (1990).

(平成4年10月14日受付)



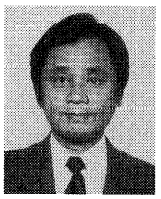
桜井成一朗（正会員）

1962年生．1989年東京工業大学大学院工学研究科博士課程修了．工学博士．同年4月から東京工業大学大学院総合理工学研究科助手．これまで，人工知能特に機械学習や類推などの高次推論の研究に従事．また法的推論にも興味をもつ．人工知能学会，日本ソフトウェア科学会，認知科学会各会員．



脇園　竜次（正会員）

1966年生．1988年鹿児島大学理学部数学科卒業．1990年九州大学大学院総合理工学研究科情報システム学専攻修士課程修了．同年(株)東芝入社．現在，同社重電技術研究所に勤務．情報処理システムの研究開発に従事．



原尾　政輝（正会員）

1944年生．1972年東北大学大学院工学研究科博士課程修了．工学博士．同年4月から東北大学電気通信研究所助手，1976年同助教授．1980～1982年フンボルト奨学生（ブラウンシュバイク工科大学客員研究員）．1984年山形大学工学部教授．1989年4月から九州工業大学情報工学部教授．これまで，セルラーオートマトンとアルゴリズム理論，並列処理理論，人工知能基礎論の研究に従事．現在，論理に基づく知能ソフトウェア理論の研究などに興味をもっている．人工知能学会，電子情報通信学会，日本ソフトウェア科学会各会員．